# CSC321 Lecture 19: Boltzmann Machines

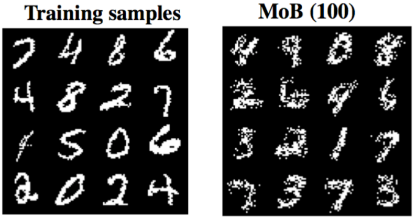
Roger Grosse

# Overview

- Last time: fitting mixture models
  - This is a kind of localist representation: each data point is explained by exactly one category
  - Distributed representations are much more powerful.
- Today, we'll talk about a different kind of latent variable model, called Boltzmann machines.
  - It's a kind of distributed representation.
  - The idea is to learn soft constraints between variables.

# Overview

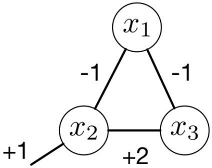
- In Assignment 4, you will fit a mixture model to images of handwritten digits.




- Problem: if you use one component per digit class, there's still lots of variability. Each component distribution would have to be really complicated.
- Some 7's have strokes through them. Should those belong to a separate mixture component?

# Boltzmann Machines

- A lot of what we know about images consists of soft constraints, e.g. that neighboring pixels probably take similar values
- A Boltzmann machine is a collection of binary random variables which are coupled through soft constraints. For now, assume they take values in $\{-1, 1\}$.
- We represent it as an undirected graph:




- The biases determine how much each unit likes to be on (i.e. $= 1$)
- The weights determine how much two units like to take the same value

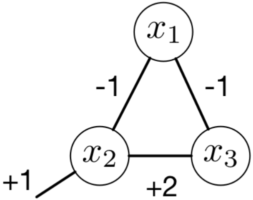
# Boltzmann Machines

- A Boltzmann machine defines a probability distribution, where the probability of any joint configuration is log-linear in a happiness function $H$.

$$p(\mathbf{x}) = \frac{1}{\mathcal{Z}} \exp(H(\mathbf{x}))$$

$$\mathcal{Z} = \sum_{\mathbf{x}} \exp(H(\mathbf{x}))$$

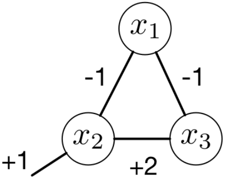
$$H(\mathbf{x}) = \sum_{i \neq j} w_{ij} x_i x_j + \sum_i b_i x_i$$




- $\mathcal{Z}$ is a normalizing constant called the partition function
- This sort of distribution is called a Boltzmann distribution, or Gibbs distribution.
  - Note: the happiness function is the negation of what physicists call the energy. Low energy = happy.
  - In this class, we'll use happiness rather than energy so that we don't have lots of minus signs everywhere.

# Boltzmann Machines

Example:



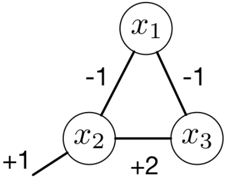

| $x_1$ | $x_2$ | $x_3$ | $w_{12}x_1x_2$ | $w_{13}x_1x_3$ | $w_{23}x_2x_3$ | $b_2x_2$ | $H(\mathbf{x})$ | $\exp(H(\mathbf{x}))$ | $p(\mathbf{x})$ |
|---|---|---|---|---|---|---|---|---|---|
| -1 | -1 | -1 | -1 | -1 | 2 | -1 | -1 | 0.368 | 0.0021 |
| -1 | -1 | 1 | -1 | 1 | -2 | -1 | -3 | 0.050 | 0.0003 |
| -1 | 1 | -1 | 1 | -1 | -2 | 1 | -3 | 0.368 | 0.0021 |
| -1 | 1 | 1 | 1 | 1 | 2 | 1 | 5 | 148.413 | 0.8608 |
| 1 | -1 | -1 | 1 | 1 | 2 | -1 | 3 | 20.086 | 0.1165 |
| 1 | -1 | 1 | 1 | -1 | -2 | -1 | -3 | 0.050 | 0.0003 |
| 1 | 1 | -1 | -1 | 1 | -2 | 1 | -1 | 0.368 | 0.0021 |
| 1 | 1 | 1 | -1 | -1 | 2 | 1 | 1 | 2.718 | 0.0158 |

$$\mathcal{Z} = 172.420$$

# Boltzmann Machines

Marginal probabilities:

$$p(x_1 = 1) = \frac{1}{\mathcal{Z}} \sum_{\mathbf{x}: x_1 = 1} \exp(H(\mathbf{x}))$$
$$= \frac{20.086 + 0.050 + 0.368 + 2.718}{172.420}$$
$$= 0.135$$




| $x_1$ | $x_2$ | $x_3$ | $w_{12}x_1x_2$ | $w_{13}x_1x_3$ | $w_{23}x_2x_3$ | $b_2x_2$ | $H(\mathbf{x})$ | $\exp(H(\mathbf{x}))$ | $p(\mathbf{x})$ |
|---|---|---|---|---|---|---|---|---|---|
| -1 | -1 | -1 | -1 | -1 | 2 | -1 | -1 | 0.368 | 0.0021 |
| -1 | -1 | 1 | -1 | 1 | -2 | -1 | -3 | 0.050 | 0.0003 |
| -1 | 1 | -1 | 1 | -1 | -2 | 1 | -3 | 0.368 | 0.0021 |
| -1 | 1 | 1 | 1 | 1 | 2 | 1 | 5 | 148.413 | 0.8608 |
| 1 | -1 | -1 | 1 | 1 | 2 | -1 | 3 | 20.086 | 0.1165 |
| 1 | -1 | 1 | 1 | -1 | -2 | -1 | -3 | 0.050 | 0.0003 |
| 1 | 1 | -1 | -1 | 1 | -2 | 1 | -1 | 0.368 | 0.0021 |
| 1 | 1 | 1 | -1 | -1 | 2 | 1 | 1 | 2.718 | 0.0158 |

$$\mathcal{Z} = 172.420$$

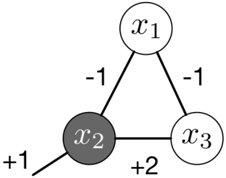
# Boltzmann Machines

Conditional probabilities:

$$p(x_1 = 1 \mid x_2 = -1) = \frac{\sum_{\mathbf{x}:x_1=1,x_2=-1} \exp(H(\mathbf{x}))}{\sum_{\mathbf{x}:x_2=-1} \exp(H(\mathbf{x}))}$$

$$= \frac{20.086 + 0.050}{0.368 + 0.050 + 20.086 + 0.050}$$

$$= 0.980$$




| $x_1$ | $x_2$ | $x_3$ | $w_{12}x_1x_2$ | $w_{13}x_1x_3$ | $w_{23}x_2x_3$ | $b_2x_2$ | $H(\mathbf{x})$ | $\exp(H(\mathbf{x}))$ | $p(\mathbf{x})$ |
|---|---|---|---|---|---|---|---|---|---|
| -1 | -1 | -1 | -1 | -1 | 2 | -1 | -1 | 0.368 | 0.0021 |
| -1 | -1 | 1 | -1 | 1 | -2 | -1 | -3 | 0.050 | 0.0003 |
| -1 | 1 | -1 | 1 | -1 | -2 | 1 | -3 | 0.368 | 0.0021 |
| -1 | 1 | 1 | 1 | 1 | 2 | 1 | 5 | 148.413 | 0.8608 |
| 1 | -1 | -1 | 1 | 1 | 2 | -1 | 3 | 20.086 | 0.1165 |
| 1 | -1 | 1 | 1 | -1 | -2 | -1 | -3 | 0.050 | 0.0003 |
| 1 | 1 | -1 | -1 | 1 | -2 | 1 | -1 | 0.368 | 0.0021 |
| 1 | 1 | 1 | -1 | -1 | 2 | 1 | 1 | 2.718 | 0.0158 |

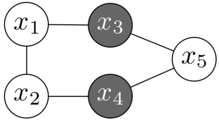
# Boltzmann Machines

- We just saw conceptually how to compute:
  - the partition function $\mathcal{Z}$
  - the probability of a configuration, $p(\mathbf{x}) = \exp(H(\mathbf{x}))/\mathcal{Z}$
  - the marginal probability $p(x_i)$
  - the conditional probability $p(x_i \mid x_j)$
- But these brute force strategies are impractical, since they require summing over exponentially many configurations!
- For those of you who have taken complexity theory: these tasks are #P-hard.
- Two ideas which can make the computations more practical
  - Obtain approximate samples from the model using Gibbs sampling
  - Design the pattern of connections to make inference easy

# Conditional Independence

- Two sets of random variables $\mathcal{X}$ and $\mathcal{Y}$ are conditionally independent given a third set $\mathcal{Z}$ if they are independent under the conditional distribution given values of $\mathcal{Z}$.
- Example:

$$
\begin{aligned}
&p(x_1, x_2, x_5 \mid x_3, x_4) \\
&\propto \exp\left(w_{12}x_1x_2 + w_{13}x_1x_3 + w_{24}x_2x_4 + w_{35}x_3x_5 + w_{45}x_4x_5\right) \\
&= \underbrace{\exp\left(w_{12}x_1x_2 + w_{13}x_1x_3 + w_{24}x_2x_4\right)}_{\text{only depends on } x_1,\, x_2} \underbrace{\exp\left(w_{35}x_3x_5 + w_{45}x_4x_5\right)}_{\text{only depends on } x_5}
\end{aligned}
$$

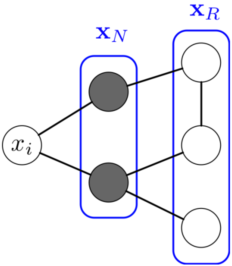



- In this case, $x_1$ and $x_2$ are conditionally independent of $x_5$ given $x_3$ and $x_4$.
- In general, two random variables are conditionally independent if they are in disconnected components of the graph when the observed nodes are removed.
- This is covered in much more detail in CSC 412.

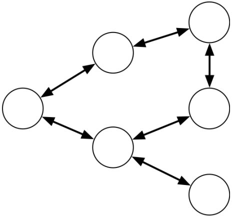
# Conditional Probabilities

- We can compute the conditional probability of $x_i$ given its neighbors in the graph.
- For this formula, it's convenient to make the variables take values in $\{0, 1\}$, rather than $\{-1, 1\}$.
- Formula for the conditionals (derivation in the lecture notes):

$$\begin{aligned}\Pr(x_i = 1 \,|\, \mathbf{x}_N, \mathbf{x}_R) &= \Pr(x_i = 1 \,|\, \mathbf{x}_N) \\ &= \sigma\left(\sum_{j \in N} w_{ij} x_j + b_i\right)\end{aligned}$$




- Note that it doesn't matter whether we condition on $\mathbf{x}_R$ or what its values are.
- This is the same as the formula for the activations in an MLP with logistic units.
  - For this reason, Boltzmann machines are sometimes drawn with bidirectional arrows.



# Gibbs Sampling

- Consider the following process, called Gibbs sampling
- We cycle through all the units in the network, and sample each one from its conditional distribution given the other units:

$$\Pr(x_i = 1 \,|\, \mathbf{x}_{-i}) = \sigma\left(\sum_{j \neq i} w_{ij} x_j + b_i\right)$$

- It's possible to show that if you run this procedure long enough, the configurations will be distributed approximately according to the model distribution.
- Hence, we can run Gibbs sampling for a long time, and treat the configurations like samples from the model
- To sample from the conditional distribution $p(x_i \,|\, \mathbf{x}_A)$, for some set $\mathbf{x}_A$, simply run Gibbs sampling with the variables in $\mathbf{x}_A$ clamped

# Learning a Boltzmann Machine

- A Boltzmann machine is parameterized by weights and biases, just like a neural net.
- So far, we've taken these for granted. How can we learn them?
- For now, suppose all the units correspond to observables (e.g. image pixels), and we have a training set $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$.
- Log-likelihood:

$$\begin{aligned}
\ell &= \frac{1}{N} \sum_{i=1}^{N} \log p(\mathbf{x}^{(i)}) \\
&= \frac{1}{N} \sum_{i=1}^{N} [H(\mathbf{x}^{(i)}) - \log \mathcal{Z}] \\
&= \left[ \frac{1}{N} \sum_{i=1}^{N} H(\mathbf{x}^{(i)}) \right] - \log \mathcal{Z}
\end{aligned}$$

- Want to increase the average happiness and decrease $\log \mathcal{Z}$

# Learning a Boltzmann Machine

- Derivatives of average happiness:

$$\begin{aligned}
\frac{\partial}{\partial w_{jk}} \frac{1}{N} \sum_i H(\mathbf{x}^{(i)}) &= \frac{1}{N} \sum_i \frac{\partial}{\partial w_{jk}} H(\mathbf{x}^{(i)}) \\
&= \frac{1}{N} \sum_i \frac{\partial}{\partial w_{jk}} \left[ \sum_{j' \neq k'} w_{j',k'} x_{j'} x_{k'} + \sum_{j'} b_{j'} x_{j'} \right] \\
&= \frac{1}{N} \sum_i x_j x_k \\
&= \mathbb{E}_{\mathrm{data}}[x_j x_k]
\end{aligned}$$

## Learning a Boltzmann Machine

- Derivatives of $\log \mathcal{Z}$:

$$\begin{aligned}
\frac{\partial}{\partial w_{jk}} \log \mathcal{Z} &= \frac{\partial}{\partial w_{jk}} \log \sum_{\mathbf{x}} \exp(H(\mathbf{x})) \\
&= \frac{\frac{\partial}{\partial w_{jk}} \sum_{\mathbf{x}} \exp(H(\mathbf{x}))}{\sum_{\mathbf{x}} \exp(H(\mathbf{x}))} \\
&= \frac{\sum_{\mathbf{x}} \exp(H(\mathbf{x})) \frac{\partial}{\partial w_{jk}} H(\mathbf{x})}{\mathcal{Z}} \\
&= \sum_{\mathbf{x}} p(\mathbf{x}) \frac{\partial}{\partial w_{jk}} H(\mathbf{x}) \\
&= \sum_{\mathbf{x}} p(\mathbf{x}) x_j x_k \\
&= \mathbb{E}_{\text{model}}[x_j x_k]
\end{aligned}$$

# Learning a Boltzmann Machine

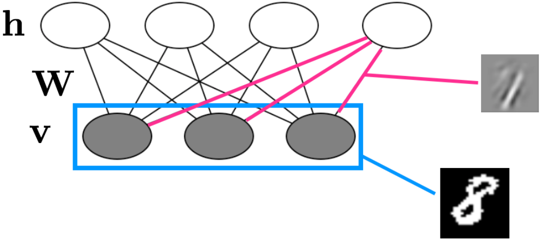
- Putting this together:

$$\frac{\partial \ell}{\partial w_{jk}} = \mathbb{E}_{\mathrm{data}}[x_j x_k] - \mathbb{E}_{\mathrm{model}}[x_j x_k]$$

- Intuition: if $x_j$ and $x_k$ co-activate more often in the data than in samples from the model, then increase the weight to make them co-activate more often.
- The two terms are called the positive and negative statistics
- Can estimate $\mathbb{E}_{\mathrm{data}}[x_j x_k]$ stochastically using mini-batches
- Can estimate $\mathbb{E}_{\mathrm{model}}[x_j x_k]$ by running a long Gibbs chain

# Restricted Boltzmann Machines

- We've assumed the Boltzmann machine was fully observed. But more commonly, we'll have hidden units as well.
- A classic architecture called the restricted Boltzmann machine assumes a bipartite graph over the visible units and hidden units:




- We would like the hidden units to learn more abstract features of the data.

# Restricted Boltzmann Machines

- Our maximum likelihood update rule generalizes to the case of unobserved variables (derivation in the notes)

$$\frac{\partial \ell}{\partial w_{jk}} = \mathbb{E}_{\mathrm{data}}[v_j h_k] - \mathbb{E}_{\mathrm{model}}[v_j h_k]$$

- Here, the data distribution refers to the conditional distribution given $\mathbf{v}$

$$\mathbb{E}_{\mathrm{data}}[v_j h_k] = \frac{1}{N} \sum_{i=1}^{N} v_j^{(i)} \mathbb{E}[h_k \mid \mathbf{v}^{(i)}]$$

- We're filling in the hidden variables using their posterior expectations, just like in E-M!

# Restricted Boltzmann Machines

- Under the bipartite structure, the hidden units are all conditionally independent given the visibles, and vice versa:
- Since the units are independent, we can vectorize the computations just like for MLPs:

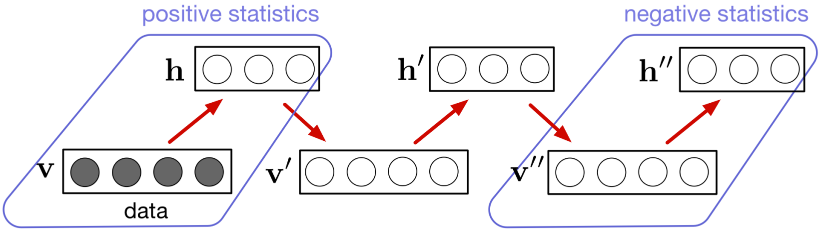
$$\tilde{\mathbf{h}} = \mathbb{E}[\mathbf{h} \mid \mathbf{v}] = \sigma\left(\mathbf{W}\mathbf{v} + \mathbf{b}_{\mathbf{h}}\right)$$
$$\tilde{\mathbf{v}} = \mathbb{E}[\mathbf{v} \mid \mathbf{h}] = \sigma\left(\mathbf{W}^\top\mathbf{h} + \mathbf{b}_{\mathbf{v}}\right)$$

- Vectorized updates:

$$\frac{\partial \ell}{\partial \mathbf{W}} = \mathbb{E}_{\mathbf{v}\sim\mathrm{data}}[\tilde{\mathbf{h}}\mathbf{v}^\top] - \mathbb{E}_{\mathbf{v},\mathbf{h}\sim\mathrm{model}}[\mathbf{h}\mathbf{v}^\top]$$

# Restricted Boltzmann Machines

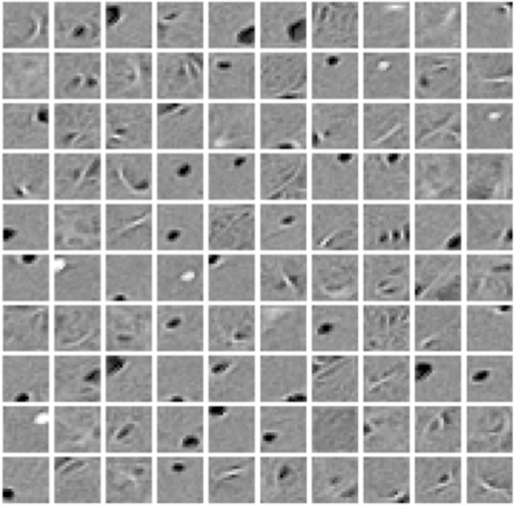
- To estimate the model statistics for the negative update, start from the data and run a few steps of Gibbs sampling.
- By the conditional independence property, all the hiddens can be sampled in parallel, and then all the visibles can be sampled in parallel.




- This procedure is called contrastive divergence.
- It's a terrible approximation to the model distribution, but it appears to work well anyway.

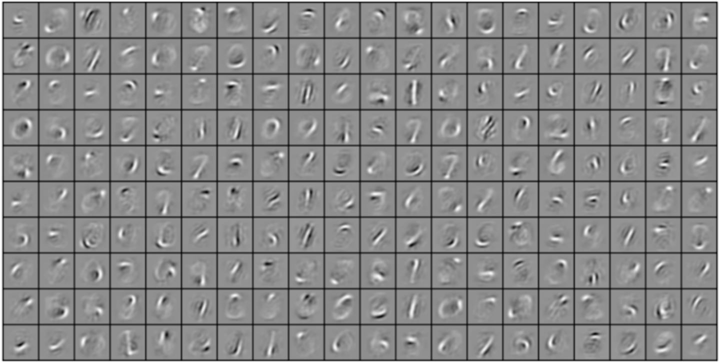
# Restricted Boltzmann Machines

Some features learned by an RBM on MNIST:



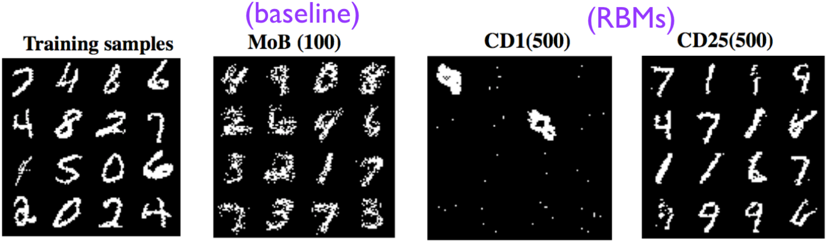
# Restricted Boltzmann Machines

Some features learned on MNIST with an additional sparsity constraint (so that each hidden unit activates only rarely):



# Restricted Boltzmann Machines

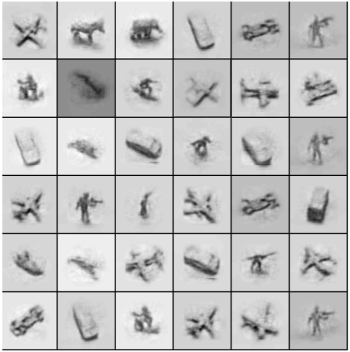
- RBMs vs. mixture of Bernoullis as generative models of MNIST



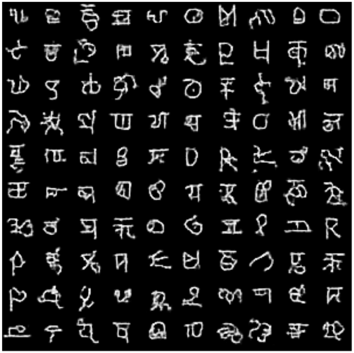

- Log-likelihood scores on the test set:
  - MoB: -137.64 nats
  - RBM: -86.34 nats
  - 50 nat difference!

# Restricted Boltzmann Machines

- Other complex datasets that Boltzmann machines can model:



NORB (action figures)



Omniglot (characters in many world languages)